Parameter Tying and Parameter Sharing

Sargur N. Srihari
srihari@cedar.buffalo.edu

Regularization Strategies

1. Parameter Norm Penalties
2. Norm Penalties as Constrained Optimization
3. Regularization and Under-constrained Problems
4. Data Set Augmentation
5. Noise Robustness
6. Semi-supervised learning
7. Multi-task learning
8. Early Stopping
9. Parameter tying and parameter sharing
10. Sparse representations
11. Bagging and other ensemble methods
12. Dropout
13. Adversarial training
14. Tangent methods

Topics in Parameter Tying/Sharing

1. Other methods for prior knowledge of parameters
2. Parameter Tying
3. Parameter Sharing
4. Parameter sharing in CNNs

Another expression for parameter prior

- L^2 regularization (or weight decay) penalizes model parameters for deviating from fixed value of zero
- Sometimes we need other ways to express prior knowledge of parameters
- We may know from domain and model architecture that there should be some dependencies between model parameters

Parameter Tying

- We want to express that certain parameters should be close to one another

A scenario of parameter tying

- Two models performing the same classification task (with same set of classes) but with somewhat different input distributions
- Model A with parameters $\boldsymbol{w}^{(A)}$
- Model B with parameters $\boldsymbol{w}^{(B)}$
- The two models map the input to two different but related outputs

$$\hat{y}^{(A)} = f(\boldsymbol{w}^{(A)}, \boldsymbol{x})$$
$$\hat{y}^{(B)} = g(\boldsymbol{w}^{(B)}, \boldsymbol{x})$$

L^2 penalty for parameter tying

- If the tasks are similar enough (perhaps with similar input and output distributions) then we believe that the model parameters should be close to each other:

$$\forall i,\ w_i^{(A)} \approx w_i^{(B)}$$

- We can leverage this information via regularization
- Use a parameter norm penalty

$$\Omega(\boldsymbol{w}^{(A)}, \boldsymbol{w}^{(B)}) = \|\boldsymbol{w}^{(A)} - \boldsymbol{w}^{(B)}\|_2^2$$

Use of parameter tying

- Approach was used for regularizing the parameters of one model, trained as a supervised classifier, to be close to the parameters of another model, trained in an unsupervised paradigm (to capture the distribution of the input data)
  - Ex. of unsupervised learning: k-means clustering
    - Input $\boldsymbol{x}$ is mapped to a one-hot vector $\boldsymbol{h}$. If $\boldsymbol{x}$ belongs to cluster i then $h_i=1$ and rest are zero corresponding to its cluster
      - It could trained using an autoencoder with k hidden units

Parameter Sharing

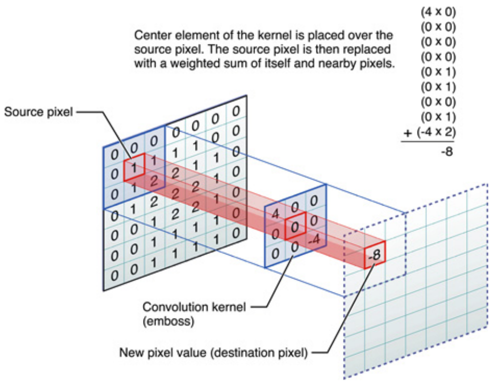
- Parameter sharing forces sets of parameters to be equal
- Because we interpret various models or model components as sharing a unique set of parameters
- Only a subset of the parameters needs to be stored in memory
  - In a CNN significant reduction in the memory footprint of the model

Use of parameter sharing in CNNs

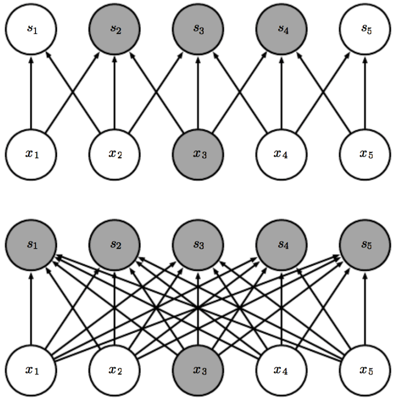
- Most extensive use of parameter sharing is in convolutional neural networks (CNNs)
- Natural images have many statistical properties that are invariant to translation
  - Ex: photo of a cat remains a photo of a cat if it is translated one pixel to the right
  - CNNs take this property into account by sharing parameters across multiple image locations
  - Thus we can find a cat with the same cat detector whether the cat appears at column i or column $i+1$ in the image

Simple description of CNN

Convolution operation

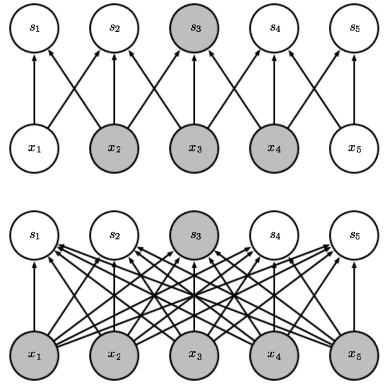


Sparsity viewed from below



- Highlight one input x_3 and output units s affected by it
- *Top*: when s is formed by convolution with a kernel of width 3, only three outputs are affected by x_3
- *Bottom:* when s is formed by matrix multiplication connectivity is no longer sparse
  - So all outputs are affected by x_3

Sparsity viewed from above



- Highlight one output s_3 and
- inputs x that affect this unit
  - These units are known as the receptive field of s_3